I claim:

1. A method of compiling a source language routine, the method comprising:

a. generating in a computer system an intermediate language routine from the source language routine;

b. specifying an initial value of each routine variable;

c. performing an optimizing change to the intermediate language routine that results in an altered intermediate language routine;

d. generating a machine language routine in a computer system from the altered intermediate language routine;

e. initializing the variables to the specified initial value;

f. executing the machine language routine from a main memory of a first computing system having the architecture of a target computer system using the initialized values;

g. measuring a characteristic of the execution; and

h. evaluating whether a stopping criterion after said executing is met and if not, repeating said performing through said measuring, saving the machine language routine having a best measured characteristic, until the stopping criterion is met.

2. The method defined in claim 1 further including before said performing:

generating a machine language routine in a computer system from the intermediate language routine;

executing the machine language routine from the main memory of the first computing system using the initialized values; and

Sub E1

measuring a characteristic of the execution.

3. The method defined in claim 1 wherein said characteristic includes at least one of a timing wherein the best measured timing is a lowest timing, a machine language routine size, and a bus utilization metric.

4. The method defined in claim 1, further including defining a plurality of segments within the intermediate language routine, each said segment comprising consecutive intermediate language routine statements wherein no segment includes a same intermediate language routine statement, and the performing an optimizing change is performed within one of the segments.

5. The method defined in claim 1 further including determining ordering dependencies in said intermediate language routine wherein said performing an optimizing change includes maintaining the determined ordering dependencies.

6. The method defined in claim 1 wherein the optimizing change comprises one of a generic optimization, a reordering, a user selectable reordering, a user selectable global reordering, a user selectable insertion of at least one instruction in a selectable position in the intermediate language routine, and a user selectable removal of at least one instruction from a selectable position in the intermediate language routine; wherein each optimizing change does not affect the intermediate language routine integrity.

Sub a1

7. The method defined in claim 1 further including after the generating the machine language routine and before the executing the machine language routine, at least one user selectable optimization to the machine language routine.

8. The method defined in claim 1 wherein the optimizing changes in a sequence of a plural number of a repeated said performing resulting from the stopping criterion not met is performed according to a process that includes at least one of a non-repeating optimizing change, a user selectable optimization change sequence; and a parallel search across a plural number of processing units.

9. The method defined in claim 1 wherein the initializing further includes initializing the position of at least part of said machine language routine in the first computing system memory, and the executing includes executing the machine language using the initialized position.

10. A machine-readable medium that provides instructions, which when executed by a processor, cause said processor to perform operations comprising:

a. generating an intermediate language routine from a source language routine;

b. executing a measuring routine to define the measurement of a characteristic of an execution of a compiled representation of the routine;

c. performing an optimizing change to the intermediate language routine that results in an altered intermediate language routine;

d. generating an object language routine from the altered intermediate language routine;

e. initializing the routine variables to a user specified values;

[illegible]

f. measuring the characteristic of an execution of the object routine using the initialized variables and the measuring routine; and

g. evaluating whether a stopping criterion is met after said executing and if not, repeating said performing through said evaluating, saving the object language file having a best measured characteristic, until the stopping criterion is met.

11. The article defined in claim 10 wherein the operations further include before the performing:

generating an object language routine from the intermediate language routine; and measuring the characteristic of an execution of the machine language routine with the measuring routine.

12. The article defined in claim 10 wherein the characteristic includes at least one of a timing wherein the best measured timing is a lowest timing, an object code size, or a bus utilization.

13. The article defined in claim 10 wherein the operations further include defining a plurality of segments within the intermediate language routine, each said segment comprising consecutive intermediate code statements wherein no segment includes a same intermediate language routine code, and the performing an optimizing change is performed within one of the segments.

14. The article defined in claim 10 wherein the operations further include determining ordering dependencies in the intermediate language routine, wherein the performing an optimizing change includes maintaining the determined ordering dependencies.

[illegible]

~~15. The article defined in claim 10 wherein the optimizing change comprises one of a generic optimization, a reordering, a user selectable reordering, a user selectable global reordering, a user selectable insertion of at least one instruction in a selectable position in the intermediate language routine, and a user selectable removal of at least one instruction from a selectable position in the intermediate language routine; wherein each optimizing change does not affect the intermediate language file integrity.~~

~~16. The article defined in claim 10 wherein the operations further include after the generating the object language routine and before the executing, implementing at least one user selectable optimization to the object language file.~~

~~17. The article defined in claim 10 wherein the optimizing changes in a sequence of a plural number of a repeated said performing resulting from the stopping criterion not met is performed according to a process that includes at least one of a non-repeating optimizing change, a user selectable optimization change sequence; and a parallel search across a plural number of processing units.~~

~~18. The article defined in claim 10 wherein the initializing further includes initializing the position of at least a portion of the object language routine in a main memory of a computing system that performs the execution.~~

[illegible]

~~19. The article defined in claim 10 wherein the operations further include a user interface for reading from the user at least one of the specified values of the routine variables, and optimizing instructions wherein the performing operation includes implementing the optimizing instructions.~~

~~20. An apparatus comprising:~~

~~a. a compiler to translate a source routine into an intermediate routine, and translate an intermediate routine into an object routine;~~

~~b. a file initializer to initialize each variable of an object routine to an initial value according to a user specified value;~~

~~c. an execution characteristic measurer to measure a characteristic of an execution of the object routine;~~

~~d. an optimizer to implement an optimizing change in the intermediate routine; and~~

~~e. a computing system having an architecture of a target computing system to execute the optimized object file from a main memory of the computing system in which the variables are assigned the initial values, and to execute the execution characteristic measurer.~~

~~21. The apparatus defined in claim 20 further including an evaluator to determine if the characteristic of an execution of the optimized object routine is in accordance with a stopping criterion and if not, to repeat an execution of the optimizer, an execution of the re-optimized code, and an execution of the execution characteristic measurer, until the stopping criterion is met.~~

Sub C1

22. The apparatus defined in claim 20 wherein the optimizer further includes in a sequence of repeated executions of the optimizer, and execution of the optimized object code, at least one of a non-repeating optimizing change, and a user selectable optimization change sequence.

23. The apparatus defined in claim 20 wherein the computing system executes an object file that is not optimized in the target computing system from the main memory, that includes the initialized variables and an execution of the execution characteristic measurer.

24. The apparatus defined in claim 20 wherein the characteristic includes at least one of a timing wherein the best measured timing is the lowest timing, a machine language size, and a bus utilization.

25. The apparatus defined in claim 20 further including a code segmenter to determine a plurality of segments within the intermediate routine, each segment comprising consecutive intermediate routine statements wherein no segment includes a same intermediate language statement, and the optimizer implements a change to one of the segments.

26. The apparatus defined in claim 20 wherein the optimizing change includes a determination of ordering dependencies in the intermediate file and the optimizing change maintains the determined ordering dependencies.

27. The apparatus defined in claim 20 wherein the optimizing change includes one of a generic optimization, a reordering, a user selectable reordering, a user selectable global

Sub a1

reordering, a user selectable insertion of at least one instruction in a selectable position in the intermediate language routine, and a user selectable removal of at least one instruction from a selectable position in the intermediate language routine; and wherein each optimizing change does not affect the intermediate file integrity.

28. The apparatus defined in claim 20 wherein the change in the intermediate routine file includes a user selectable optimization.

29. The apparatus defined in claim 20 wherein the file initializer is further to initialize the position of at least a portion of the object routine file in the main memory, and the computing system is further to allocate the main memory according to the initialized positions of the portion of the object routine.

30. The apparatus defined in claim 20 wherein the computing system includes a plurality of processors that each have an architecture of the target computing system.